

Exercises

Exercise 8.1 In less than one page, describe four everyday activities that exhibit temporal or spatial locality. List two activities for each type of locality, and be specific.

Exercise 8.2 In one paragraph, describe two short computer applications that exhibit temporal and/or spatial locality. Describe how. Be specific.

Exercise 8.3 Come up with a sequence of addresses for which a direct mapped cache with a size (capacity) of 16 words and block size of 4 words outperforms a fully associative cache with least recently used (LRU) replacement that has the same capacity and block size.

Exercise 8.4 Repeat [Exercise 8.3](#) for the case when the fully associative cache outperforms the direct mapped cache.

Exercise 8.5 Describe the trade-offs of increasing each of the following cache parameters while keeping the others the same:

- (a) block size
- (b) associativity
- (c) cache size

Exercise 8.6 Is the miss rate of a two-way set associative cache always, usually, occasionally, or never better than that of a direct mapped cache of the same capacity and block size? Explain.

Exercise 8.7 Each of the following statements pertains to the miss rate of caches. Mark each statement as true or false. Briefly explain your reasoning; present a counterexample if the statement is false.

- (a) A two-way set associative cache always has a lower miss rate than a direct mapped cache with the same block size and total capacity.
- (b) A 16-KB direct mapped cache always has a lower miss rate than an 8-KB direct mapped cache with the same block size.
- (c) An instruction cache with a 32-byte block size usually has a lower miss rate than an instruction cache with an 8-byte block size, given the same degree of associativity and total capacity.

Exercise 8.8 A cache has the following parameters: b , block size given in numbers of words; S , number of sets; N , number of ways; and A , number of address bits.

- (a) In terms of the parameters described, what is the cache capacity, C ?
- (b) In terms of the parameters described, what is the total number of bits required to store the tags?
- (c) What are S and N for a fully associative cache of capacity C words with block size b ?
- (d) What is S for a direct mapped cache of size C words and block size b ?

Exercise 8.9 A 16-word cache has the parameters given in [Exercise 8.8](#). Consider the following repeating sequence of 1_w addresses (given in hexadecimal):

40 44 48 4C 70 74 78 7C 80 84 88 8C 90 94 98 9C 0 4 8 C 10 14 18 1C 20

Assuming least recently used (LRU) replacement for associative caches, determine the effective miss rate if the sequence is input to the following caches, ignoring startup effects (i.e., compulsory misses).

- (a) direct mapped cache, $b = 1$ word
- (b) fully associative cache, $b = 1$ word
- (c) two-way set associative cache, $b = 1$ word
- (d) direct mapped cache, $b = 2$ words

Exercise 8.10 Repeat Exercise 8.9 for the following repeating sequence of 1_w addresses (given in hexadecimal) and cache configurations. The cache capacity is still 16 words.

74 A0 78 38C AC 84 88 8C 7C 34 38 13C 388 18C

- (a) direct mapped cache, $b = 1$ word
- (b) fully associative cache, $b = 2$ words
- (c) two-way set associative cache, $b = 2$ words
- (d) direct mapped cache, $b = 4$ words

Exercise 8.11 Suppose you are running a program with the following data access pattern. The pattern is executed only once.

0x0 0x8 0x10 0x18 0x20 0x28

- (a) If you use a direct mapped cache with a cache size of 1 KB and a block size of 8 bytes (2 words), how many sets are in the cache?
- (b) With the same cache and block size as in part (a), what is the miss rate of the direct mapped cache for the given memory access pattern?
- (c) For the given memory access pattern, which of the following would decrease the miss rate the most? (Cache capacity is kept constant.) Circle one.
 - (i) Increasing the degree of associativity to 2.
 - (ii) Increasing the block size to 16 bytes.
 - (iii) Either (i) or (ii).
 - (iv) Neither (i) nor (ii).

Exercise 8.12 You are building an instruction cache for a MIPS processor. It has a total capacity of $4C = 2^{c+2}$ bytes. It is $N = 2^n$ -way set associative ($N \geq 8$), with a block size of $b = 2^{b'}$ bytes ($b \geq 8$). Give your answers to the following questions in terms of these parameters.

- (a) Which bits of the address are used to select a word within a block?
- (b) Which bits of the address are used to select the set within the cache?
- (c) How many bits are in each tag?
- (d) How many tag bits are in the entire cache?

Exercise 8.13 Consider a cache with the following parameters:

N (associativity) = 2, b (block size) = 2 words, W (word size) = 32 bits, C (cache size) = 32 K words, A (address size) = 32 bits. You need consider only word addresses.

- (a) Show the tag, set, block offset, and byte offset bits of the address. State how many bits are needed for each field.
- (b) What is the size of *all* the cache tags in bits?
- (c) Suppose each cache block also has a valid bit (V) and a dirty bit (D). What is the size of each cache set, including data, tag, and status bits?
- (d) Design the cache using the building blocks in Figure 8.78 and a small number of two-input logic gates. The cache design must include tag storage, data

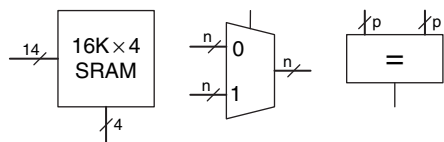


Figure 8.78 Building blocks

storage, address comparison, data output selection, and any other parts you feel are relevant. Note that the multiplexer and comparator blocks may be any size (n or p bits wide, respectively), but the SRAM blocks must be $16K \times 4$ bits. Be sure to include a neatly labeled block diagram. You need only design the cache for reads.

Exercise 8.14 You’ve joined a hot new Internet startup to build wrist watches with a built-in pager and Web browser. It uses an embedded processor with a multilevel cache scheme depicted in [Figure 8.79](#). The processor includes a small on-chip cache in addition to a large off-chip second-level cache. (Yes, the watch weighs 3 pounds, but you should see it surf!)

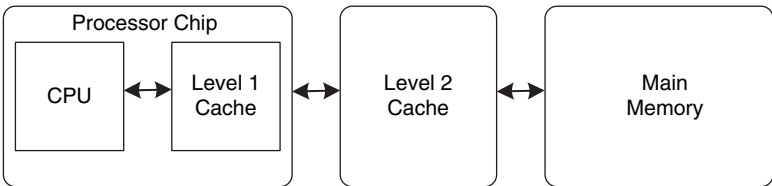


Figure 8.79 Computer system

Assume that the processor uses 32-bit physical addresses but accesses data only on word boundaries. The caches have the characteristics given in [Table 8.14](#). The DRAM has an access time of t_m and a size of 512 MB.

Table 8.14 Memory characteristics

Characteristic	On-chip Cache	Off-chip Cache
Organization	Four-way set associative	Direct mapped
Hit rate	A	B
Access time	t_a	t_b
Block size	16 bytes	16 bytes
Number of blocks	512	256K

- (a) For a given word in memory, what is the total number of locations in which it might be found in the on-chip cache and in the second-level cache?
- (b) What is the size, in bits, of each tag for the on-chip cache and the second-level cache?
- (c) Give an expression for the average memory read access time. The caches are accessed in sequence.
- (d) Measurements show that, for a particular problem of interest, the on-chip cache hit rate is 85% and the second-level cache hit rate is 90%. However, when the on-chip cache is disabled, the second-level cache hit rate shoots up to 98.5%. Give a brief explanation of this behavior.

Exercise 8.15 This chapter described the least recently used (LRU) replacement policy for multiway associative caches. Other, less common, replacement policies include first-in-first-out (FIFO) and random policies. FIFO replacement evicts the block that has been there the longest, regardless of how recently it was accessed. Random replacement randomly picks a block to evict.

- (a) Discuss the advantages and disadvantages of each of these replacement policies.
- (b) Describe a data access pattern for which FIFO would perform better than LRU.

Exercise 8.16 You are building a computer with a hierarchical memory system that consists of separate instruction and data caches followed by main memory. You are using the MIPS multicycle processor from [Figure 7.41](#) running at 1 GHz.

- (a) Suppose the instruction cache is perfect (i.e., always hits) but the data cache has a 5% miss rate. On a cache miss, the processor stalls for 60 ns to access main memory, then resumes normal operation. Taking cache misses into account, what is the average memory access time?
- (b) How many clock cycles per instruction (CPI) on average are required for load and store word instructions considering the non-ideal memory system?
- (c) Consider the benchmark application of [Example 7.7](#) that has 25% loads, 10% stores, 11% branches, 2% jumps, and 52% R-type instructions.⁶ Taking the non-ideal memory system into account, what is the average CPI for this benchmark?
- (d) Now suppose that the instruction cache is also non-ideal and has a 7% miss rate. What is the average CPI for the benchmark in part (c)? Take into account both instruction and data cache misses.

⁶ Data from Patterson and Hennessy, *Computer Organization and Design*, 4th Edition, Morgan Kaufmann, 2011. Used with permission.

Exercise 8.17 Repeat Exercise 8.16 with the following parameters.

- (a) The instruction cache is perfect (i.e., always hits) but the data cache has a 15% miss rate. On a cache miss, the processor stalls for 200 ns to access main memory, then resumes normal operation. Taking cache misses into account, what is the average memory access time?
- (b) How many clock cycles per instruction (CPI) on average are required for load and store word instructions considering the non-ideal memory system?
- (c) Consider the benchmark application of [Example 7.7](#) that has 25% loads, 10% stores, 11% branches, 2% jumps, and 52% R-type instructions. Taking the non-ideal memory system into account, what is the average CPI for this benchmark?
- (d) Now suppose that the instruction cache is also non-ideal and has a 10% miss rate. What is the average CPI for the benchmark in part (c)? Take into account both instruction and data cache misses.

Exercise 8.18 If a computer uses 64-bit virtual addresses, how much virtual memory can it access? Note that 2^{40} bytes = 1 *terabyte*, 2^{50} bytes = 1 *petabyte*, and 2^{60} bytes = 1 *exabyte*.

Exercise 8.19 A supercomputer designer chooses to spend \$1 million on DRAM and the same amount on hard disks for virtual memory. Using the prices from [Figure 8.4](#), how much physical and virtual memory will the computer have? How many bits of physical and virtual addresses are necessary to access this memory?

Exercise 8.20 Consider a virtual memory system that can address a total of 2^{32} bytes. You have unlimited hard drive space, but are limited to only 8 MB of semiconductor (physical) memory. Assume that virtual and physical pages are each 4 KB in size.

- (a) How many bits is the physical address?
- (b) What is the maximum number of virtual pages in the system?
- (c) How many physical pages are in the system?
- (d) How many bits are the virtual and physical page numbers?
- (e) Suppose that you come up with a direct mapped scheme that maps virtual pages to physical pages. The mapping uses the least significant bits of the virtual page number to determine the physical page number. How many virtual pages are mapped to each physical page? Why is this “direct mapping” a bad plan?
- (f) Clearly, a more flexible and dynamic scheme for translating virtual addresses into physical addresses is required than the one described in part (e). Suppose

you use a page table to store mappings (translations from virtual page number to physical page number). How many page table entries will the page table contain?

- (g) Assume that, in addition to the physical page number, each page table entry also contains some status information in the form of a valid bit (*V*) and a dirty bit (*D*). How many bytes long is each page table entry? (Round up to an integer number of bytes.)
- (h) Sketch the layout of the page table. What is the total size of the page table in bytes?

Exercise 8.21 Consider a virtual memory system that can address a total of 2^{50} bytes. You have unlimited hard drive space, but are limited to 2 GB of semiconductor (physical) memory. Assume that virtual and physical pages are each 4 KB in size.

- (a) How many bits is the physical address?
- (b) What is the maximum number of virtual pages in the system?
- (c) How many physical pages are in the system?
- (d) How many bits are the virtual and physical page numbers?
- (e) How many page table entries will the page table contain?
- (f) Assume that, in addition to the physical page number, each page table entry also contains some status information in the form of a valid bit (*V*) and a dirty bit (*D*). How many bytes long is each page table entry? (Round up to an integer number of bytes.)
- (g) Sketch the layout of the page table. What is the total size of the page table in bytes?

Exercise 8.22 You decide to speed up the virtual memory system of [Exercise 8.20](#) by using a translation lookaside buffer (TLB). Suppose your memory system has the characteristics shown in [Table 8.15](#). The TLB and cache miss rates indicate

Table 8.15 Memory characteristics

Memory Unit	Access Time (Cycles)	Miss Rate
TLB	1	0.05%
Cache	1	2%
Main memory	100	0.0003%
Hard drive	1,000,000	0%

how often the requested entry is not found. The main memory miss rate indicates how often page faults occur.

- (a) What is the average memory access time of the virtual memory system before and after adding the TLB? Assume that the page table is always resident in physical memory and is never held in the data cache.
- (b) If the TLB has 64 entries, how big (in bits) is the TLB? Give numbers for data (physical page number), tag (virtual page number), and valid bits of each entry. Show your work clearly.
- (c) Sketch the TLB. Clearly label all fields and dimensions.
- (d) What size SRAM would you need to build the TLB described in part (c)? Give your answer in terms of depth \times width.

Exercise 8.23 You decide to speed up the virtual memory system of [Exercise 8.21](#) by using a translation lookaside buffer (TLB) with 128 entries.

- (a) How big (in bits) is the TLB? Give numbers for data (physical page number), tag (virtual page number), and valid bits of each entry. Show your work clearly.
- (b) Sketch the TLB. Clearly label all fields and dimensions.
- (c) What size SRAM would you need to build the TLB described in part (b)? Give your answer in terms of depth \times width.

Exercise 8.24 Suppose the MIPS multicycle processor described in [Section 7.4](#) uses a virtual memory system.

- (a) Sketch the location of the TLB in the multicycle processor schematic.
- (b) Describe how adding a TLB affects processor performance.

Exercise 8.25 The virtual memory system you are designing uses a single-level page table built from dedicated hardware (SRAM and associated logic). It supports 25-bit virtual addresses, 22-bit physical addresses, and 2^{16} -byte (64 KB) pages. Each page table entry contains a physical page number, a valid bit (V), and a dirty bit (D).

- (a) What is the total size of the page table, in bits?
- (b) The operating system team proposes reducing the page size from 64 to 16 KB, but the hardware engineers on your team object on the grounds of added hardware cost. Explain their objection.

- (c) The page table is to be integrated on the processor chip, along with the on-chip cache. The on-chip cache deals only with physical (not virtual) addresses. Is it possible to access the appropriate set of the on-chip cache concurrently with the page table access for a given memory access? Explain briefly the relationship that is necessary for concurrent access to the cache set and page table entry.
- (d) Is it possible to perform the tag comparison in the on-chip cache concurrently with the page table access for a given memory access? Explain briefly.

Exercise 8.26 Describe a scenario in which the virtual memory system might affect how an application is written. Be sure to include a discussion of how the page size and physical memory size affect the performance of the application.

Exercise 8.27 Suppose you own a personal computer (PC) that uses 32-bit virtual addresses.

- (a) What is the maximum amount of virtual memory space each program can use?
- (b) How does the size of your PC's hard drive affect performance?
- (c) How does the size of your PC's physical memory affect performance?

Exercise 8.28 Use MIPS memory-mapped I/O to interact with a user. Each time the user presses a button, a pattern of your choice displays on five light-emitting diodes (LEDs). Suppose the input button is mapped to address 0xFFFFF10 and the LEDs are mapped to address 0xFFFFF14. When the button is pushed, its output is 1; otherwise it is 0.

- (a) Write MIPS code to implement this functionality.
- (b) Draw a schematic for this memory-mapped I/O system.
- (c) Write HDL code to implement the address decoder for your memory-mapped I/O system.

Exercise 8.29 Finite state machines (FSMs), like the ones you built in [Chapter 3](#), can also be implemented in software.

- (a) Implement the traffic light FSM from [Figure 3.25](#) using MIPS assembly code. The inputs (T_A and T_B) are memory-mapped to bit 1 and bit 0, respectively, of address 0xFFFFF000. The two 3-bit outputs (L_A and L_B) are mapped to bits 0–2 and bits 3–5, respectively, of address 0xFFFFF004. Assume one-hot output encodings for each light, L_A and L_B ; red is 100, yellow is 010, and green is 001.

- (b) Draw a schematic for this memory-mapped I/O system.
- (c) Write HDL code to implement the address decoder for your memory-mapped I/O system.

Exercise 8.30 Repeat [Exercise 8.29](#) for the FSM in [Figure 3.30\(a\)](#). The input *A* and output *Y* are memory-mapped to bits 0 and 1, respectively, of address 0xFFFFF040.